```
import pandas as pd
```

```
df = pd.read_csv('/kaggle/input/sales-data/kaggle sale.csv')
print(df.head())
```

```
     User ID  Gender  Age  EstimatedSalary  Purchased satisfied
0  15624510    Male   19            19000          0         no
1  15810944    Male   35            20000          0         no
2  15668575  Female   26            43000          0         no
3  15603246  Female   27            57000          0         no
4  15804002    Male   19            76000          0         no
```

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   User ID          400 non-null    int64 
 1   Gender           400 non-null    object
 2   Age              400 non-null    int64 
 3   EstimatedSalary  400 non-null    int64 
 4   Purchased        400 non-null    int64 
 5   satisfied        400 non-null    object
dtypes: int64(4), object(2)
memory usage: 18.9+ KB
None
```

```
df = df.drop(columns = ['User ID'])
print(df.head())
```

```
   Gender  Age  EstimatedSalary  Purchased satisfied
0    Male   19            19000          0         no
1    Male   35            20000          0         no
2  Female   26            43000          0         no
3  Female   27            57000          0         no
4    Male   19            76000          0         no
```

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
col = ['Gender','satisfied ']
for i in col:
    df[i] = le.fit_transform(df[i])
```

```
print(df.head())
```

```
   Gender  Age  EstimatedSalary  Purchased  satisfied
0       1   19            19000          0          0
1       1   35            20000          0          0
2       0   26            43000          0          0
3       0   27            57000          0          0
4       1   19            76000          0          0
```

```
x = df[['Gender','Age','EstimatedSalary','Purchased']]
y = df['satisfied ']
```

```
print(x.head())
```

```
   Gender  Age  EstimatedSalary  Purchased
0       1   19            19000          0
1       1   35            20000          0
2       0   26            43000          0
3       0   27            57000          0
4       1   19            76000          0
```

```
print(y.head())
```

```
0    0
1    0
2    0
3    0
4    0
Name: satisfied , dtype: int64
```

```
from sklearn.model_selection import train_test_split
```

```
x_train, x_test, y_train, y_test = train_test_split(x,y,random_state=42)
```

```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(300, 4)
(300,)
(100, 4)
(100,)
```

```
from sklearn.ensemble import RandomForestClassifier
```

```
rfc = RandomForestClassifier(n_estimators=100, criterion='gini',max_features='sqrt',random_stat
```

```
rfc.fit(x_train,y_train)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
prediction = rfc.predict(x_test)
```

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
print("Random Forest Classifier Accuracy:", accuracy_score(y_test, prediction))
```

```
Random Forest Classifier Accuracy: 0.74
```

```
print("Classification Report:\n", classification_report(y_test, prediction))
```

```
Classification Report:
                precision    recall  f1-score   support

           0       0.59      0.55      0.57        31
           1       0.80      0.83      0.81        69

    accuracy                           0.74       100
   macro avg       0.69      0.69      0.69       100
weighted avg       0.74      0.74      0.74       100
```

```
print("Confusion Matrix:\n", confusion_matrix(y_test, prediction))
```

```
Confusion Matrix:
 [[17 14]
 [12 57]]
```

try with entropy as criterion

```
rfc_2 = RandomForestClassifier(n_estimators=100, criterion='entropy',max_features='sqrt',random
```

```
rfc_2.fit(x_train,y_train)
```

```
RandomForestClassifier(criterion='entropy', random_state=42)
```

```
pred = rfc_2.predict(x_test)
```

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
print("Random Forest Classifier Accuracy:", accuracy_score(y_test, pred))
```

```
Random Forest Classifier Accuracy: 0.75
```

```
print("Classification Report:\n", classification_report(y_test, pred))
```

```
Classification Report:
                precision    recall  f1-score   support

           0       0.61      0.55      0.58        31
           1       0.81      0.84      0.82        69

    accuracy                           0.75       100
   macro avg       0.71      0.69      0.70       100
weighted avg       0.74      0.75      0.75       100
```

```
print("Confusion Matrix:\n", confusion_matrix(y_test, pred))
```

```
Confusion Matrix:
 [[17 14]
 [11 58]]
```